COMPUTER
SIGNAL
PROCESSORS, INC.

To:

T. A. Nelson
Systems Consultant
Box 3
Schooleys Mountain, New Jersey
07870

209 Middlesex Turnpike, Burlington, Massachusetts 01803

csp COMPUTER SIGNAL PROCESSORS, INC.

PHONE (617) 272-6020

209 Middlesex Turnpike, Burlington, Massachusetts 01803

TWX (710) 332-1352

13 May 1970

Dear Mr. Nelson:

Thank you for your continuing expression of interest and technical curiosity about the world's fastest 16-bit computer: the CSP-30.

The enclosed booklet describes in considerable detail such useful information as physical specifications, operating parameters, and includes a detailed description of the instruction set. I am sure that somewhere in the booklet you will find the answer to almost any question. If not, please give us a call and we will be more than happy to provide answers of application descriptions as they apply to your situation.

We look forward to hearing from you.

Very truly yours,

COMPUTER SIGNAL PROCESSORS, INC.

R. D. Stout
Director of Sales

jrw

Encl. 040120

# The company the products & the people

cspi COMPUTER SIGNAL PROCESSORS, INC.

# The company....

*Computer Signal Processors, Inc. is the first company to be established for the specific purpose of designing and manufacturing digital systems for signal processing.*

Founded in 1968, the company quickly outgrew its initial quarters in Waltham, Massachusetts. Early in 1969, the home office and manufacturing facilities were transferred to newly constructed quarters in nearby Burlington. This ideal location has made it possible to draw upon a body of skill and talent almost unique to this geographic area as well as affording access to the wealth of special knowledge and developmental techniques available in the greater metropolitan Boston area.

Atypically for a firm of its type, CSPI's growth has been marked by the early achievement of many organizationally important goals. Meticulous initial planning has resulted in a balance of corporate capabilities supported by reputable service organizations. The anticipation of needs has also enabled the company to staff with solid competence in all necessary areas: Advanced Design, Production Engineering, System Programming, Applications Engineering, Product Assurance, Documentation, Business Administration, Marketing, and Sales.

*Equally important have been the technical milestones:*

- Well within a year of the company's founding, its scientists and engineers had designed and successfully produced its first system product, the CompuSignal System-3.
- They had developed and satisfactorily delivered signal processing systems answering special user requirements.
- They had conceptualized and completed the system design of a second major product for signal processing.
- They had realized all these achievements within very reasonable tolerances of the planned budgets and schedules.

# the products....

The company's initial product is a fast, flexible, complete signal processing system, the CSS-3. This low-cost system features complete time-series analysis software for real time Fast Fourier Transforms, Cepstra, Convolutions, Correlations, Filters, Signal Averages, and Histograms; an analog sampling-converting input; a programmed 16-bit digital processor-memory; two 10-bit digital-analog output channels; sampling to 50 kHz; teletypewriter, tape reader and punch for input and output, an x-y display oscilloscope, and a control console and working desk. There are also many optional software and hardware features that make CSS-3 an exceptionally flexible system.

The system is presently in production and will be offered on an off-the-shelf basis with delivery extensions required only for the incorporation of options or for the preparation of unusual software.

The company's second product is a signal processing device which promises to be more than an order of magnitude faster than comparable existing equipment. This product is scheduled for release before 1970.

CSPI has also undertaken the design and development of special systems for either single or multiple signal processing tasks. Such systems are conventionally produced for quantity (OEM) purchasers.

Digital signal processing systems, such as the types produced by CSPI, find applications in an ever-broadening family of sciences. A handful, for which standard and special programming has been developed, includes:

Communications Research
Electronic Intelligence
Speech Studies
Sonar Research
Oceanographic Research
Geological Exploration
Chemical Analysis
Medical Research
Acoustics Research
Biochemical Measurements
Vibration Analysis
Radio Astronomy
Seismology
Shock Studies

# the people.

*CSPI's founders, under the guidance of the distinguished scientist, Dr. Edmund U. Cohler, have pioneered the application of digital techniques to the field of signal processing since long before the company's formal incorporation.*

**Dr. Cohler,** *CSPI's President,* was previously a Senior Scientist at Sylvania's Applied Research Laboratory where he directed computer research. During 1967 and 1968 he formed a group whose aim was to translate a family of concepts into an entirely new computer: the ACP-1, one of the first specialized digital signal processors. At least partly as a result of this successful undertaking, the group became the nucleus of CSPI.

**Michael M. Stern,** *Vice President of Operations,* was previously Director of Engineering for Di/An Controls, Inc., where he directed and coordinated all new product design and development, advanced techniques evaluation, technical proposal efforts, and reliability analysis and assurance. Mr. Stern's knowledge of digital systems and memory design and his skills in the area of quality equipment manufacture resulted in the country's most outstanding production system of aerospace memories.

**Dr. Donald N. Graham,** *Director of Software and Systems Analysis,* came to CSPI from Sylvania's Applied Research Laboratory. There, Dr. Graham developed a number of software systems for digital signal processing, computer-driven display, alphanumeric display, and handprinted character recognition. He has recently developed several new algorithms for high-speed digital spectral analysis and digital demodulation of signals.

**James A. Waggett,** *Director of Advanced Computer Design,* joined Dr. Cohler's computer design group at Sylvania in 1966. Mr. Waggett designed the logic for the ACP-1, supervised its testing, and wrote programs for analyzing the nature of unfamiliar communications signals.

**Andrew E. Emery,** *Manager of Marketing and Promotion,* came to CSPI from Avco where his previous experience as Marketing VP for a California computer systems firm enabled him to provide marketing support for the EDP and N/C activities of two divisions. Mr. Emery also served as a management analyst for Avco's CSPCS system.

**Robert D. Stout,** *National Sales Manager,* is a recent arrival from Scientific Data Systems, Inc., where he was primarily instrumental in structuring and managing all sales activities in a tightly competitive region. He brings to CSPI a fresh range of management skills and an intimate knowledge of the problem-solving techniques of the field.

*The company also benefits from its expanding staff of reputable scientists, as well as the guidance available from such eminent advisors as its Board of Directors, which includes:*

**Professor W. J. Poppelbaum,** of the University of Illinois, has been active for many years in the development of techniques for Illiac computers at the University's Digital Computer Laboratory.

**R. Willis Leith, Jr.,** a Senior Partner of Boston's brokerage firm of Burgess & Leith and a former president of the Boston Stock Exchange, is a notable expert in the field of equity financing.

**Walter W. Kenyon,** formerly associated with Millipore Corp., has a history of singular success in the guidance of small businesses.

**Darley T. Randall,** a Vice President of Fiduciary Trust Company of New York, is a recognized expert in banking and general fiscal knowledge.

## The future of Digital Signal Processing

Until recently, computer signal processing amounted to little more than a few special digital filters for frequencies so low that inductors and capacitors were impractically large. Then Fast Fourier Transforms and low-cost digital integrated circuits were introduced. This combination, plus development, has made digital signal processing available to science and industry.

The response has been dramatic. As the results of research become available and as the development of specific algorithms continue, applications proliferate faster than equipment. In this atmosphere, CSPI is convinced that digital signal processing will grow even faster than the larger field of general data processing.


209 Middlesex Turnpike, Burlington, Massachusetts 01803 • (617) 272-6020

089009

Printed in U.S.A.

# THE CSP-30

## AN ULTRA-FAST DIGITAL COMPUTER

csp i COMPUTER SIGNAL PROCESSORS, INC.

# TABLE OF CONTENTS

# TABLE OF CONTENTS (cont'd.)

# LIST OF FIGURES

FIGURE NO.

Figure 1-1 The CSP-30 High Speed Digital Computer

# I INTRODUCTION

## 1.1 FEATURES

The CSP-30 is an extremely fast, 16-bit, general purpose digital computer with an unusually powerful and sophisticated instruction set. Among the more noteworthy features that contribute to its high processing speed, its practicality, its breadth of application, and its economy of operation are:

- Combination Memory. An integrated circuit (IC) memory and a core memory are combined to optimize the balance between speed and economy. Either memory can hold instructions and data.

- Fast IC Memory. The IC memory has a complete basic cycle time of 100 nanoseconds. This results in instruction times of 100 nanoseconds and up; typically, 300 nanoseconds, or over 3 million instructions per second.

- Fast Core Memory. Although the core has a full cycle time of 900 nanoseconds, a unique "background fetch" allows the core latency time to be ignored, so the core can look only 100 nanoseconds slower than the IC memory, or only 200 nanoseconds in many problems.

- Direct Memory Access (DMA.) Two optional DMA ports, one for each memory, are available. The core port can operate independently of the IC memory and computer, allowing read or write word-rates up to 1.4 MHz without decreasing the computational capability. The IC DMA allows read or write word-rates of 10 MHz.

- Accumulators. 32 accumulators are available to the programmer, of which 15 may be used as index registers, permitting dedication of accumulators to fast interrupt routines and minimizing overhead time.

- Hardware Multiply. The CSP-30 includes a hardware multiply with a speed of 1.0 microsecond.

- Arithmetic Instructions. Arithmetic and logical commands specify the source of two operands and the destination of results. They replace the usual 3-instruction FETCH-OPERATE STORE sequence with a single instruction.

- Automatic "Push And Pop" Instructions. These make it possible to do re-entrant subroutines and interrupts with a minimum of overhead. The complete overhead for an interrupt is 1.2 microseconds.

- Complete Arithmetic & Logical Test Instructions. These permit testing for the set or reset condition of any combinations of bits, or for the relative state of an index.

- Connect Instructions. These unique I/O instructions allow a word from a selected peripheral device to be used as an operand in the next instruction (connect input). They also allow the result of the next instruction to be transferred to a selected output device (connect output).

- Input/Output Subsystem. The I/O subsystem includes both highspeed and party-line I/O channels.

- Priority Interrupts. The I/O subsystem provides for multiple, multilevel priority interrupts, which may be either externally-initiated or program-initiated. The programmer provides enabling control for each device

- I/O Status Information. Each I/O channel contains status information features that allow simple interfacing with any real time or conventional peripheral.

Every CSP-30 user has additional access to CSPI's impressive and growing library of applications programs and processing algorithms, except in certain proprietary areas. This library is the result of CSPI's deep and continuing involvement in the complexities of real time digital processing and time-series analysis. The company has been instrumental in solving many basic applications problems and in meeting a variety of hardware systems needs in this field, and is now pleased to make these tools available to CSP-30 users.

# II HARDWARE DESCRIPTION

## 2.1 DESIGN COMMENTS

The CSP-30 is a totally new computer of extremely advanced design. It is the result of an unusual blend of vitally important factors, considered at every decision-making state:

- Solid experience and success in the development of high speed digital computers and systems;
- A pragmatic understanding of the most useful state-of-the-art advances in electronic components, logic design, architecture, packaging, and production techniques;
- A body of practical applications information; and
- A true appreciation of the need for human engineering and the users' actual working requirements.

Overall cleanness of the CSP-30 results from the use of such advanced elements as: MSI devices, system-wide minimum path design, multi-layer printed circuit boards based on an unusual concept of design unification, and a combination of mother-board and ultra-high reliability wire-wrap interconnections on a single back plane.

Reliability is enhanced by the use of thoroughly tested and use-proved integrated circuit components and logic modules, and by the establishment of a full Quality Assurance system following MIL-Q-9859A guidelines.

Maintainability: the ability to perform routine and emergency maintenance in the most efficient manner, is designed into the CSP-30 from detailed component to complete system level. Many features contribute to this ease of maintenance, such as a completely unified and modular design, and a simplicity of working access to all modules and component. Removing the front panel exposes the entire back plane, which in turn can be swung forward and down to provide total access to all printed circuit boards. The boards are guided into connection position, held in place, or safely and easily removed by a system of jack-screws.

The basic CSP-30, as shown in Figure 1-2, is 68" high, 21" wide, 27" deep, and weighs approximately 475 pounds, exclusive of the Teletype I/O device. Power requirements are 20 amps of 115 VAC ± 10% at 60 Hz. It will function in temperatures ranging from 0° C to 50°C, and from 20% to 90% relative humidity. No special air conditioning or sub-flooring is required.

## 2.2 THE PROCESSOR

The following descriptions and specifications of the processors's features and elements may be more easily related to each other by referring to Figure 2-2: CSP-30 Block Diagram.

### 2.2.1 Memories

These are two logically equivalent memories embodied in the same addressing structure: the integrated circuit (IC) memory and the core memory. Each memory is capable of storing both data and instructions. The most important differences between the two are speed and capacity.

Memory Specifications

| | IC | CORE |
|---|---|---|
| Full cycle time (nanosec): | 100 | 900 |
| Access time (nanosec.): | 100 | 350 |
| Half-cycle time (nanosec): | - | 600 |
| Transfer rate (million 16-bit words/sec.): | 10 | 1 |
| Transfer rate (million 8-bit words/sec.): | 20 | 2 |
| Basic memory capacity (words); | 512 | 4,096 |
| Expandable to (words): | 2,048 | 32,768 |
| In blocks of (words): | 128 | 4,096 |

*Direct memory access from or to external devices.


Figure 2-1. The CSP-30, Typical Multi-Layer CPU Printed Circuit Board

The optional core memory expansion may be accomplished on-site, but requires CSPI's Core Memory Field Expansion Option (Model 3020) which includes: additional back panel nest wiring to a cable card location; a cable card wired to the new core memory expansion plane; the mechanical structure to house either 6 or 8 additional core memory modules; and cooling, in the form of blowers, for the additional memory modules. The additional memory is physically located within the vertical equipment rack directly behind the main frame nest, thus adding no external elements to the computer. If IC memory expansion to 2048 words is anticipated, a special main frame and the Model 3020 Expander must be purchased initially.

### 2.2.2 Instructions

The CSP-30 has 291 basic instructions: 212 single-word instructions, and 91 double word instructions. There are many operational variations, resulting in an instruction set of great power. The groups are:

Arithmetic/Logical
Skip
Jump
Shift
Pushdown List & Subroutine Calls
Move/Load/Store
Input/Output

A more detailed description of the instruction set is to be found in Section V of this book.

### 2.2.3 Arithmetic & Logical

Arithmetic and logical instructions may be direct or indirect and/or indexed. They employ: 16-bit operands; fixed-point binary operations, with provisions for double precision; 2's complement form; multiply & device have a 2-word result mode; and instructions specify operand sources and result destinations.

#### Minimum Operating Times

| | |
|---|---|
| Add/Subtract: | 200 nanosec. |
| Jump: | 200 nanosec. |
| Multiply: | 1.0 microsec. |
| Divide: | 3.5 microsec. |

#### For Instruction Modification, Add:

| | |
|---|---|
| Core memory reference (not in background): | 400 nsec. read<br>100 nsec. write |
| Index: | 100 nsec. |
| IC indirect address (per level): | 100 nsec. |
| Core indirect address (per level): | 900 nsec. |

Typical instruction rate: over 3-million instructions per second.

### 2.2.4 Addressing

There are many addressing possibilities, including; direct addressing over the full range of IC and core memories; indexed addressing; multi-level indirect addressing; immediate addressing; literal addressing; and relative addressing.

### 2.2.5 Registers

The greatest number of registers is found in the accumulator file which has 32 registers, the low-order 15 of which may be used as index registers.

| | Quantity | Length (Bits) | Available * |
|---|---|---|---|
| Accumulator File: | 32 | 16 | yes |
| (quantity includes Index Registers): | (15) | 16 | yes |
| Arithmetic (A) Register: | 1 | 16 | yes |
| Memory Address (IC & Core): | 2 | 16 | no |
| Accumulator File Address: | 1 | 5 | no |
| Memory Buffer: | 1 | 16 | no |
| Instruction: | 1 | 16 | no |
| Program Counter: | 1 | 15 | no |
| Intermediate Result Arithmetic (B & C): | 2 | 17 | no |

* To Programmer

See Figure 2-2 - Block Diagram on the following page.

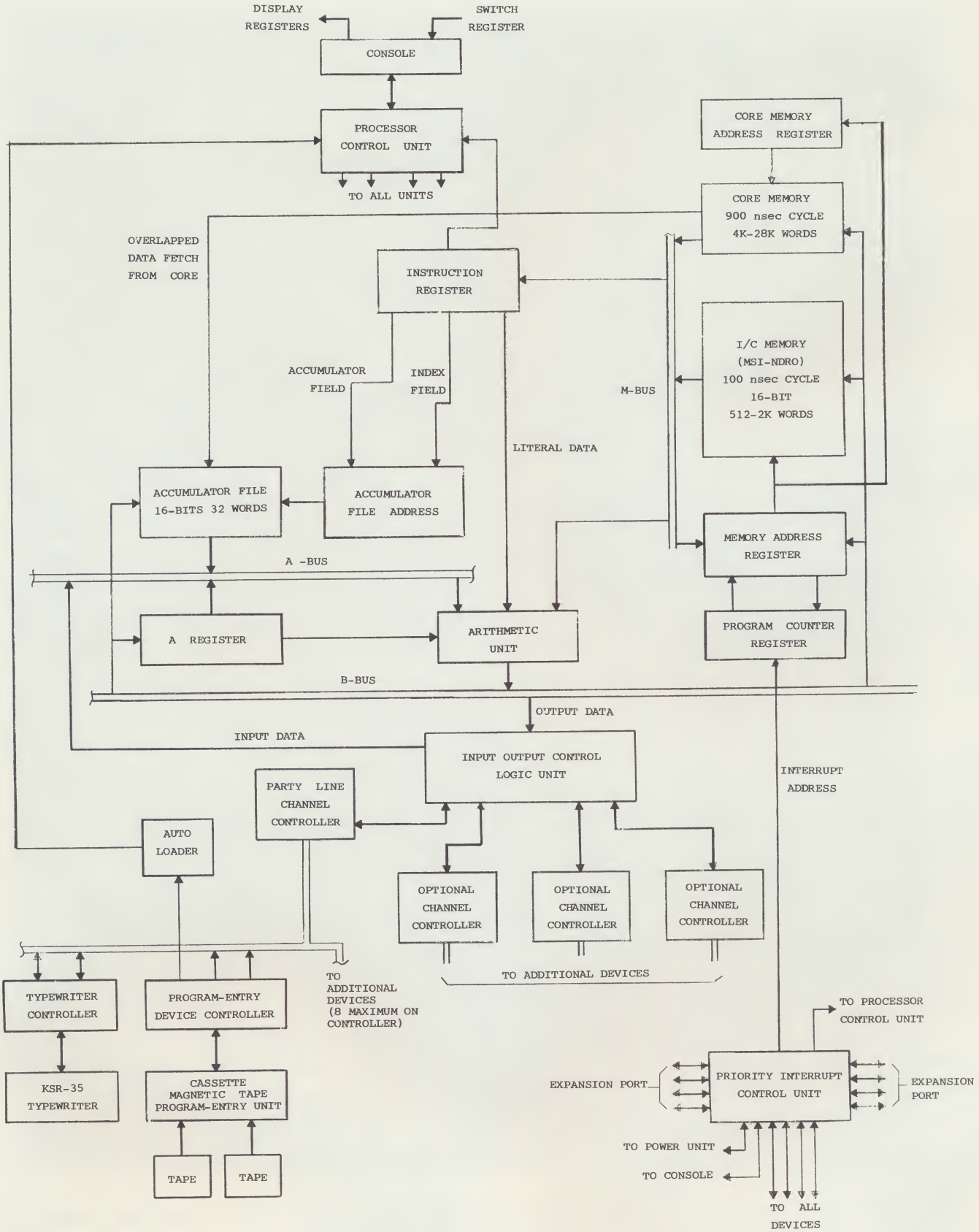



Figure 2-2. CSP-30 Block Diagram

### 2.2.6 Input/Output (I/O)

The I/O Control Logic Unit handles all direct I/O data transfers between the CPU and the peripheral devices attached to the I/O channels. It accepts commands from the CPU for transmitting data, and command codes to the peripheral controllers. It handles all data transfer and addressing chores so that the program is virtually independent of both the peripheral device structure and the transmission system.

#### I/O Features

3 parallel, expandable I/O channels for high speed devices or con - trollers; for data transfer rates in excess of 1 million words per second, i.e.: high speed disks, or analog-to-digital converters.

1 I/O channel with an 8-device party line controller ; practical for devices having medium data transfer rates: under 100,000 words per second. This channel includes:

1 address pre-assigned to magnetic tape program entry,

1 address pre-assigned to I/O teletypewriter,

6 address available for assignment.

Programmed, single-word I/O data transfers.

Programmed sensing of external devices' status.

### 2.2.7 Priority Interrupts

The CSP-30 provides a multi-level priority interrupt capability. The interrupt control unit allows an external device to interrupt the program flow and transfer control to an interrupt routine. The address of the interrupt routine is found in a memory location assigned to each interrupt level. The processor automatically stores the program counter, the contents of the A Register, and a third word defining the machine's present status. An interrupt control instruction lets the programmer inhibit each priority level or device, or initiate his own interrupt. Another instruction returns the computer to its normal pre-interrupt state.

7 interrupt levels are standard, with the following set pre-assignments:

1 high priority, pre-assigned to "power threshold" sensor.

1 pre-assigned to the control panel's INTERRUPT Switch.

4 available for assignment.

1 available for all I/O devices.

An option (Model 3060) permits the expansion of the priority interrupt control unit by the addition of one or two 6-level interrupt blocks.

## 2.3 CONTROL PANEL

The CSP-30's control panel provides interface between the operator and the system, a facility that is extremely useful for programming and maintenance. The individual controls and indicators listed and described below are number-keyed to the Control Panel illustration.

Latch On Compare/Off/Halt On Compare (1) is a 3-position switch used largely for testing and monitoring. In the Halt On Compare position it compares the 16-bit word pre-set on the Memory Address/Switch Register (21) with the contents of a selected register while running the program. When a positive correspondence occurs, the current instruction is completed and the program is halted. In the Latch On Compare position it compares in the same way as in the Halt On Compare. When a positive correspondence occurs the Memory (25) indicators are locked up until the next comparison.

Interrupt (2) interrupts the program. It is usually used for convenient exit to a debugging or other utility routine. Return from an interrupt is under software control.

Power: Off/On/Lock (3) has a built-in time delay for proper shut-down. Lock leaves the power on an prevents re-setting any control on the panel. The key may be removed at any setting: Off, On, or Lock.

See Figure 2-3. CSP-30 Control Panel on the following page.

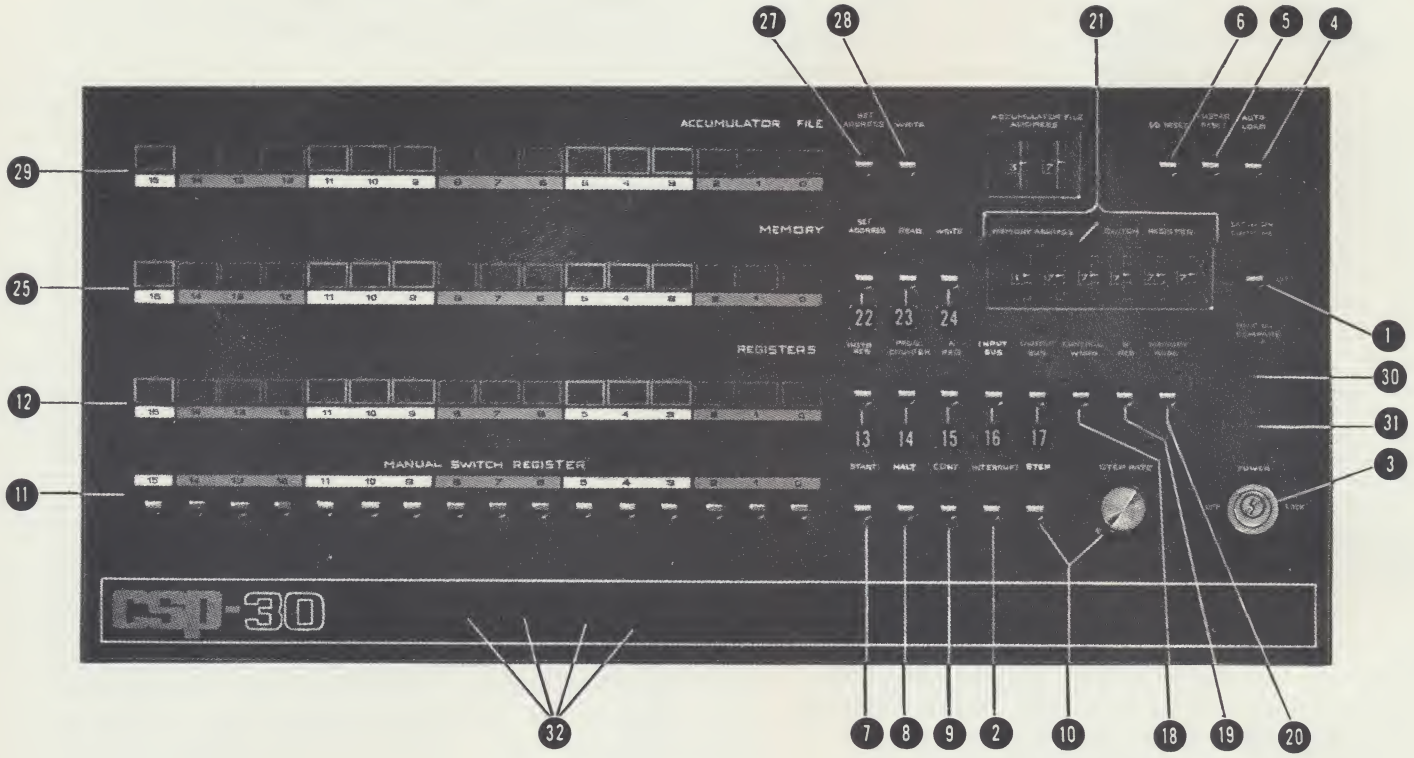



Figure 2-3. CSP-30 Control Panel

Master Reset (4) resets all registers, counters, control flops, I/O devices, etc.

Auto Load (5) causes autoloading of simplified format. Usually used for reading the program load routine into memory.

I/O Reset (6) clears all registers, resets error flip-flops, and clears interrupt-enable control conditions.

Run (7) loads the setting of the Memory Address/Switch Register (21) into the Program Counter, and starts program execution.

Halt (8) halts the system after it finishes executing the current instruction.

Continue (9) re-establishes the operational status interrupted by the Halt (8) switch. Control is returned to the Program Counter and the run state is resumed.

Step/Step Rate (10) The step switch causes a single instruction to be executed when the Step Rate knob is off. At any other Step Rate position a slow, controlled step-through-instruction takes place, which permits running through programs at visual rates.

Manual Switch Register (11) is a set of sixteen switches with "binary one" up positions and "binary zero" down positions. When the CSP-30 is in the Halt mode these switches can be used to enter data into an accumulator or a memory location. In the Run mode they are used as sense switches: their settings can be transferred to an accumulator and examined by the program.

Registers (12) are a set of sixteen indicator lamps in color-coded octal subsets, used to display the contents of a selected register, bus, counter, or control word.

The register desired is selected by one of the following switches:

Instruction Register (13);

Program Counter (14);

A Register (15);

Output Bus (16);

Input Bus (17);

Test Register (18) used with the optional Maintenance Panel to display the contents of certain internal registers;

Control Word (19) containing such machine status bits as: Overflow, Carry Divide, Check, etc.

Memory Address (20);

Memory Address/Switch Register (21) is a 6-digit thumbwheel used to select a location in either IC or core memory so that its contents may be displayed on the Memory (25) indicator, or so that new data may be entered. The octal address (000000 to 177777) is set manually.

Set Address (22) transfers any address set on the Memory Address/ Switch Register (21) into the memory address register.

Read (23) causes the memory address register to advance by one. The contents at that location are displayed on the Memory (25) indicators.

Write (24) enters the data in the Manual Switch Register (11) into the location specified by the memory address register and causes the memory addressing register to advance by one and display the contents of the new location on the Memory (25) indicators

Memory (25) is a set of sixteen indicator lamps in color-coded subsets. The set displays the contents of a selected IC or core memory location.

Accumulator File Address (26) is a 2-digit thumbwheel used to select one of the 32 accumulators. The octal address (00 to 37) is set manually.

Read (27) locates the accumulator whose address has been set on the Accumulator File Address (26) thumbwheels and displays the contents on the Accumulator File (29) indicators.

Write (28) enters the data in the Manual Switch Register (11) into the currently addressed accumulator.

Accumulator File (29) is a set of sixteen indicator lamps in color-coded octal subsets, used to display the contents of a selected accumulator.

Alarm (30) is a warning sign, invisible until illuminated, to advise the operator of potential system failure from such causes as lack of adequate cooling.

Power Fail (31) is a warning sign, invisible until illuminated, to advise the operator of the previous absence of adequate system power.

Improper Address/Indirect Loop/I/O Busy/ Memory Busy (32) is a set of four problem indicator signs, normally invisible.

## 2.4 THE POWER SUPPLY

The power supply is a heavy duty, extremely reliable module designed and manufactured to CSPI's stringent specifications. It is located in the base of the CSP-30's vertical rack assembly in order to add physical stability, improve maintenance accessibility, and facilitate total system cooling.

The module provides +5 VDC, ±15 VDC, and lamp voltage to the system. Among the many interesting features are various interlock controls:

A main circuit breaker, plus an additional circuit breaker for each of the supply voltages;

Automatic memory data protection, in which any major power variation triggers a power threshold sensing switch. This may be used to initiate a program-directed transfer of volatile data from the IC memory into core;

Overvoltage and overcurrent protection on all supplies;

Undervoltage sensing on all supplies;

AC power dropout sensing;

Internally generated signals for sequencing to interrupt during both power ON and power OFF conditions; and

Additional filtering RFI, and tight regulation of transients.

## 2.5 PERIPHERAL EQUIPMENT

### 2.5.1 I/O Teletypewriter

Each basic CSP-30 is supplied with a heavy duty I/O teletypewriter: a Teletype Model KSR 35. This device may be used for communications between the operator and the computer, and for obtaining the printout of program results.

### 2.5.2 Magnetic Tape I/O Device

Also supplied with each basic CSP-30 is a 2-deck cassette-type magnetic tape program entry device. Each cassette has a capacity of 150,000 eight-bit characters. The I/O data transfer rate is 500 characters (4000 bits) per second. The tape decks are mechanically and functionally independent, but share the read/write electronics and the control logic.

The magnetic tape cassette concept offers advantages of speed and convenience over such conventional techniques as punched paper tape. It provides convenient program storage, simplicity of handling, faster program entry, the ability to assemble programs, and may also be used for receiving and storing temporary or permanent output data or other information.

## 2.6 OPTIONAL EQUIPMENT

In addition to the IC memory and core memory expansion options described in paragraph 2.2.1, CSPI offers the following optional equipment.

- Power supply expansion, for servicing options and peripherals.
- A direct memory access (DMA) port to the core memory.
- A direct memory access port to the IC memory.
- I/O DMA controllers.
- I/O device controllers.
- DMA expanders.
- Fast channel expanders.
- A 3-cassette magnetic tape program entry module.
- A line printer.
- A deluxe I/O teletypewriter.
- Analog-to-digital converters having 10-bit or 12-bit outputs.
- Digital-to-analog converters having 10-bit or 12-bit inputs.
- Sample rate counters having rates of 1 MHz, 2MHz, and 10MHz, to provide program control of the sampling rate with great accuracy, stability and resolution.
- Magnetic disk and drum storage units of various capacities.
- Dial registers, for the continuous entry and adjustment of analog parameters.
- An X-Y plotting graphic recorder.
- A storage CRT display module.
- Magnetic tape equipment (IBM compatible) of various speeds.
- A card reader.
- A real time clock.
- A paper tape reader and punch.

Other peripherals are under consideration or are being investigated for feasibility, and may eventually be offered.

# III SOFTWARE

## 3.1 DESIGN COMMENT

Speed and applications flexibility were the ultimate design criteria used throughout the CSP-30's development phase. The result has been called "a programmer's computer". It possesses the speed, the adaptability, and even the "forgivability" that makes creative programming difficult to resist. At the same time the CSP-30 and its supplied programs a working set that simplify the tasks of even the best programmer.

## 3.2 UTILITY PROGRAMS

### 3.2.1 Standard Supplied Programs

Each basic CSP-30 is shipped with a set of utility programs that is extremely flexible and complete. Included in this set are:

- Loader - a conventional program to increase flexibility.
- Symbolic Text Editor - a line-oriented program whose language is people-tailored and extremely easy to use; an exceptionally fine editor for a 4K computer.
- Debugging Aid - a small, numeric-only program with clean, simple, and flexible control language; both input and output include octal or decimal integers, fractions, and ASC II character codes; for 4K use.
- Debugging Aid - as above, plus the ability to operate with mnemonic symbols for instructions, and with user-defined symbols for memory locations.
- Stand-Alone Symbolic Assembler - a 2-pass, free format assembler with many niceties, for 4K use.
- Stand-Alone Symbolic Assembler - as above, plus provisions for defining MACRO routines and other sophisticated assembly options.
- Fortran Symbolic Assembler - so that object programs may be prepared on large batch-processing computers.
- Suitable Diagnostic and Maintenance Programs.

The magnetic tape cassette is used as the standard storage device for all programs. It is also used by the Symbolic Text Editor and the Symbolic Assembler for input and output.

### 3.2.2 Future Programs

Other utility programs are presently being developed, or are being considered for development in the near future. They will form part of CSPI's constantly growing library of programs and algorithms which continues to increase the flexibility of the CSP-30.

A signal processing debugging aid is one such future utility program. Planned for use with the optional CRT display module, this program will provide the software equivalent of hardware testing probes and equipment. It will enable the user to dynamically display the contents of one or more memory locations during the operation of a signal processing program. The display, waveform or an x-y plot, will thus serve much the same function as an oscilloscope used in an analog system.

A signal processing-oriented higher lever language is still another program planned for future availability. This will permit the user to design his program in the form of a block diagram. Primitive blocks will be assigned mnemonics which will then be used as system building blocks. The user will be able to assign parameters at the time of compiling, or may vary them dynamically. It will thus be possible to define quite complex function in this fashion.

## 3.3 FUNCTION PROGRAMS

### 3.3.1 Types of Programs Available

CSPI's corporate involvements, as well as the experience of its pertinent personnel, have tended toward applications in the field of digital signal processing. It is not surprising, therefore, that many of the programs and algorithms in the company's library were selected and developed with this kind of processing in mind.

However, signal processing, particularly as defined by CSPI, has far broader connotations and applications than are generally realized. With the advent of practical Fast Fourier Transform algorithms, and a family of other digital processing techniques, the entire field of real time signal processing has opened up. Now signal processing applications are almost as numerous as the variety of signals to be investigated.

The CSP-30's processing functions are determined completely by software. Since each application requires somewhat different external hardware and data handling techniques, this software flexibility permits the user to achieve a true "standard product" economy, despite system differences.

The following standard programs and subroutines are available for accomplishing commonly-desired functions:

- Fast Fourier Transform - Radix-2 for minimum memory requirements and Radix-4 for maximum speed.
- Zoom FFT - provides high resolution spectra over narrow bands with minimum storage requirements and high speed.
- Cepstrum
- Directly calculated Convolution, Cross-correlation, and Auto-correlation.

  Recursive Digital Filtering - first order and second order. More complex filters may be synthesized from first and second order blocks.
- Ensemble Averaging of data blocks
  - True Average
  - Decaying Average
- Amplitude Histogram

Additional programs and subroutines are in planning or development stages and will be added to CSPI's library in the future.

### 3.3.2 Fast Fourier Transform (FFT)

#### 3.3.2.1 General Description

The Fourier Transform is one of the most valuable and broadly applicable of all signal processing techniques. Its versatility makes it attractive mathematically and now that very fast algorithms have been developed for computation it is also economical. Incidentally, it is worth noting that CSPI's FFT programs are demonstratably the best available.

Each program includes a variety of selectable features. All spectral programs permit the following:

Variable block size

Variable sampling rate

Ensemble Averaging

Hanning weighting

Zero filling

The FFT algorithm permits the computation of all spectral vectors of an incoming signal block (real or complex), as may be seen in Equation (1). The vector magnitudes represent the amplitude spectrum of the input signal.

$$Y(n\ f_0) = \sum_{k=0}^{N-1} y(k\tau)e^{j2\pi f_0 k\tau n} \qquad (1)$$

where:

$y(k\tau)$ is the incoming waveform sampled at $t = k\tau$. Will be represented as $y_k$, and may be complex.

$Y(n\ f_0)$ is the spectral component at frequency $n\ f_0$, usually complex.

$f_0$ is the frequency corresponding to the analysis period; i.e., $f_0 = \frac{1}{(N\tau)}$

n is the frequency index, which can range from $\frac{-N}{2}$ to $\frac{+N}{2}$ assuming the N is a power of 2).

$\tau$ is the time interval between samples.

N is the number of samples in the block.

Application areas for the FourierTransform are very broad. Filtering may be accomplished by transforming an incoming time series or waveform; eliminating, emphasizing, or otherwise processing the desired spectral components, and then re-transforming to obtain a times series filter output. Squared magnitude of selected spectral components may be summed to obtain a measure of energy in a third-octave band. Multiplying the spectral components by the spectral components of a reference waveform or its conjugate and re-transforming, gives either the Cross-correlation or Convolution between the input and the reference function. Auto-correlation may be performed in a manner similar to Cross-correlation, except that no stored reference waveform is required; the spectral coefficients of the incoming waveform are simply squared and then re-transformed. The Cepstrum, of interest in vocoding, is obtained by re-transforming the log-spectral- magnitudes. A more classic application is the solution of partial differential equations, in which the boundary conditions are transformed to find the equation's solution.

3.3.2.2 Timing

The time required to perform the entire FFT algorithm is determined by two factors: the actual calculation time of the transform, and the associated software "housekeeping".

The calculation time is nearly proportional to $N \log_2 N$, so if the time for one value of N is known, any other can be calculated. $\text{Log}_2$ N is a slowly-varying function of N, so the calculation time varies almost linearly with N. Since the time required to take in N samples at a fixed sampling rate also varies linearly with N, the time that is available for computation and the time that is required for computation, vary almost together in N. Also since sampling rate and bandwidth are proportional, the bandwidth that can be continuously covered is approximately a constant (a slowly varying function of N) representing the machine speed, which can be used to quickly calculate the CSP-30's suitability for a particular application.

> NOTE: The examples of actual calculation times provided in later paragraphs indicate that a bandwidth coverage of approximately 30kHz can be continuously maintained.

Much higher bandwidths can be processed on stationary signals by using discontinuous block processing. Sampling a signal with a stationary spectrum can take place at a high rate for a short time, stop while the FFT is calculated, and then resume after the calculation is finished. This procedure ignores some of the input, but this is not usually critical in the case of either a stationary or slowly- changing signal. Real time spectra can be obtained in this manner over more than 5 MHz of bandwidth. The technique is also useful whenever the signal (or information of interest) only exists for a short time, and when there is no need to analyze new information until the previous calculation has been completed.

Bandwidths up to 10 GHz can be analyzed with sampling-head techniques if the signal of interest is periodic or stochastic. For example, quadrature samples of a bandpass signal may be obtained by sampling both sides of a balanced demodulator or, in the case of narrow-band signals, by taking a second sample of the signal a quarter of a period away from the first. A pair of such samples may be treated as a complex sample, and for a bandwidth B, B complex samples per

second are required. (Since the FFT algorithm is complex by nature, it handles this type of signal best.)

| Program Timing | Data In Core Memory |
|---|---|
| Transform 256 complex samples: | 6.9 msec. |
| Output 256 complex spectral samples: | .5 msec. |
| Interrupt, input complex samples and scramble sample order: 3.8 usec x 256 = | 1.0 msec. |
| | 8.4 msec. |
| | = 30 kHz |

The frequency resolution for the example above is 119 Hz for the 30 kHz band. The time required for the transform, as noted earlier, is approximately proportional to $N \log_2 N$: to increase from N = 256 to N = 1024 would represent a $\frac{4\ (10)}{8} = 5$ increase in time. The samples are collected during a period four times as long. Thus the CSP-30 will handle approximately 2 24 kHz bandwidth at a 1024 complex point resolution.

In a second example, 2N real samples can be transformed with an N-sample FFT. Here the program is a bit longer and the cosine table is double in size. The other memory requirements are the same as for N complex samples. In this case, although 2N complex frequency samples may be calculated, they are conjugate symmetric so that only N are non-redundant.

| Program Timing | Data In Core Memory |
|---|---|
| Input, Scramble, and Output Transform 512 spectral (256 complex) samples: | 8.4 msec. |
| Even/odd separates and combine: | 1.9 msec. |
| | 10.3 msec. |
| | = 25 kHz |

A similar possibility is to use two independent channels for the real and imaginary parts of a complex input. One of these channels could be the square of the previous spectrum, thus resulting in the auto-correlation function of the previous signal segment.

| Program Timing | Data In Core Memory |
|---|---|
| Input, Scramble, Transform, and Output 256 sample pairs: | 8.4 msec. |
| Even/odd separate | .1 msec. |
| | 8.5 msec/256 complex samples |
| | = 30 kHz for 2 |
| | = 30 kHz for 2 channels |
| | = 15 kHz bandwidth each |

The times given above are for the transform calculation plus housekeeping. Additional time is required if the operator calls for spectral magnitudes, cepstra, spectral estimates, or correlations, since certain other incidental processes are necessary. Below are listed the times required for these processes when results are returned to the core memory. If data is outputted the times will be approximately .5 microsecond less per sample, and the output time included above would also be included here.

| | |
|---|---|
| Spectrum Magnitude: | 3.0 μsec/complex spectral value |
| Log Spectrum: | 2.5 - 4.0 μsec/complex spectral value |
| Hanning Weight input samples (affects side lobes of spectrum coefficients): | 2.0 μsec/real sample |
| Complex Multiply (for cross-correlation) | 5.5 μsec/complex sample |

Example: Taking a cepstrum over 256 real input samples, and using the complex sample's imaginary part for the "return" transform, would require:

| | |
|---|---|
| Input, Scramble, Transform, Output | 8.4 msec. |
| Spectrum Magnitude (128 x 3 μsec) = | .4 msec. |
| Log Spectrum (128 x 3.0 μsec) = | .4 msec. |
| Hanning Weight (256 x 2.0 μsec) = | .5 msec. |
| | 9.7 msec/256 samples |

= 26 kHz sample rate, or
= 13 kHz bandwidth

3.3.2.3 Resolution

The resolution, in frequency, is the inverse of the integration time ($\frac{\text{sampling rate}}{\text{number of samples}}$) Another way to look at it is: with 2N real samples (N complex samples) the spectrum out to half the real sample rate (the complex sample rate) is divided into N pieces; thus the total spectrum is $\frac{1}{2}$ Hz for real samples ($\frac{1}{}$ Hz for complex samples), and each spectral sample is separated from adjacent samples by $\frac{1}{2N}$ Hz ($\frac{1}{N}$ for complex samples). Clearly, then, resolution is a function of N, or the amount of storage available.

The FFT algorithms, performed over N complex input samples, require the following storage:

2N words (approximately) reserved for data storage and intermediate calculations

2N words for a buffer if continuous operation is required

$\frac{N}{2}$ words (approximately for cosine tables

300 words for the program itself

External memory makes it possible to perform very large transforms. For instance, CSPI has been successful in routinely performing 32,000 - sample transforms by using the following memory combination: 512 words of IC memory, 4096 words of core memory, and a disk memory.

3.3.2.4 Errors

A frequently-expressed concern among potential users of the CSP-30 is its ability to reduce noise due to calculation errors. For the CSP-30 doing fixed-point calculations, a bound for the rms error in this spectrum may be calculated from:

$$\text{RMS Error} = 2^{(M-31)/2} \times \frac{\text{RMS Result}}{\text{RMS Input}}$$

where $M = \log_2 N$

The peak errors may be considerably above normal due to dependence of samples, and to quantization distortion aliasing. CSPI has considerable experience in the techniques of this area, and is able to offer significant assistance in the analysis and solutions of defined problems.

3.3.3 Zoom FFT

CSPI's special Zoom FFT performs a bandpass filter operation on the incoming time sequence, and then transforms a decimated version of the complex filtered output. It is most commonly used for taking a "fine grained" look at some portion of the frequency spectrum. Zooming resolves $1/2^m$ of the basic spectrum into as many frequency elements as the full spectrum. This technique permits a storage saving of $2^m$ over conventional requirements. The time required to calculate an N-point Zoom FFT is about the same as the FFT time multiplied by $2^m$ (the Zoom ratio).

### 3.3.4 Cepstrum

The Cepstrum function is a sort of modified Auto-correlation. It is obtained by Fourier transforming a time function, taking the logorithm of the Spectral Magnitudes, and inversely transforming the resulting frequency function.

The Cepstrum is an excellent pitch extractor because of its efficacy in locating the fundamental period of a modulated stream of pulses. Thus, it can be used to establish the fundamental pitch period of a human voice. When displayed, this fundamental voice pitch appears as a large peak with a generally "Christmas tree" appearance. The location of this peak on the time axis is independent of the voicing of the sound: "a-a-a-a" and "e-e-e-e" when voiced with the same pitch will create the same Cepstrum peak on the display, whereas a change in pitch while voicing a continuous "a-a-a-a" will move the peak up and down the time scale. The pitch-peak has many fewer side-lobes in the cepstrum than in the auto-correlation.

Programs have also been written which operate in conjunction with the Cepstrum Program to determine the numerical values of pitch periods, and also to give some measure of reliability to the value - based on the size of the other peaks and total voice power.

### 3.3.5 Correlation and Convolution

This set of available programs makes it possible to compute Convolution, Auto-correlation, and Cross-correlation in either of two ways: directly, in the time domain; or through the frequency domain, using the FFT algorithm. Cross-correlation and Convolution each use a stored reference. Auto-correlation by FFT can be performed for integer sample lags up to 1/2 block size.

### 3.3.6 Recursive Filter Processing

Standard available programs include cascades at single and double poles, and an integrate-and-dump (optimum match to a sine wave) which preserves the phase of the input. It is anticipated that additional filter programs will be added as application dictate.

In some cases filtering waveforms by the difference equation technique is more efficient than using the Fourier Transform technique. If the filtering or analysis process to be accomplished can be described by either a linear differential equation or by a transfer function, a corresponding difference equation may be obtained. A representative example would be the difference equation representing a bandpass integrate-and-dump circuit, given as:

$$Y_{k+1} - 2\cos(w\tau)\, Y_k - Y_{k-1} = q_k$$

For each incoming sample, $q_k$, a new value of the output variable $y_{k+1}$, can be calculated based on previous samples and previous values of $y_k$.

The response of this filter is only quasi-stable and input at frequency will cause the response to diverge. However the response of such a filter may be taken as the magnitude and phase of its oscillation at some time, T. The frequency response is then the familiar $\frac{\sin x}{x}$ function with nodes at $w = \pm 1/T$. The FFT gives the same sort of response in each of its spectral components, so if many such filters are to be realized synchronously at equally spaced intervals the FFT is an efficient technique to use. For a small number of such filters, however, or for an asynchronous set of filters, the equation above is much more efficient for obtaining the desired result.

An advantage of the difference equation technique over the Fourier Transform technique is that the data storage requirement is limited to the number of poles and zeros in the transfer function, or to the number of terms in the difference equation. Also, the number of words that must be stored for instructions and constants is considerably smaller (25, for the example).

The fastest filtering that can be done with present programs is with the integrate-and-dump filter. Rates of 2.0 $\mu$sec to 6.5 $\mu$sec per sample per filter are possible (depending on scaling, storage, and indexing requirements). At a sample rate of 10 kHz it is possible to run as many as 20 filters plus other processing functions.

Other digital filters require approximately 4.0 μsec per pole pair per sample. Thus, for instance, eight sixth-order Butterworth low-pass (3 pole pairs) filters could be run at a 10 kHz sample rate.

Because of the large number of calculations per sample, round-off errors may be more prominent when using the difference equation technique. The simplest way to check for possible problems here is to examine the stability of the difference equation. If it is unstable, or if its stability is marginal, the round-off errors appearing as continual noise perturbations to the forcing function may cause a build-up of spurious outputs at the characteristic modes of the difference equation. Thus the problem of round-off in a filter may be treated as signal-to-noise. CSPI specialists can assist in the calculation of such effects.

### 3.3.7 Ensemble: Averaging True & Decaying

The decaying average, triggered internally or externally, computes the ensemble average of blocks of data using a selectable time constant. This averaging brings a signal out of very adverse noise conditions.

The time constant may be changed to accommodate slowly varying, non-stationary signals. The averaging is accomplished by multiplying the previous value of output by 1-S and adding S times the latest sample (S being a small positive number) to obtain the new output. This has the same effect as a unity gain filter with a time constant of $\frac{1}{S}$ on each of the spectral outputs. The operator may choose the decay decrement. With this feature, for instance, sonograms may be easily plotted on an intensity modulated display.

The "true" average computes the value of:

$$Y_{K+1} = \frac{(q_k - y_k)}{k} + Y_k$$

where K is the largest value of $2^m$ in k+1

This is very nearly the same as the usual mean value:

$$Y_{k+1} = \frac{1}{k+1} (q_k - y_k) + Y_k$$

and may be calculated most rapidly. As with the mean value, it can never overflow, thus it eliminates scaling problems encountered using single cumulation.

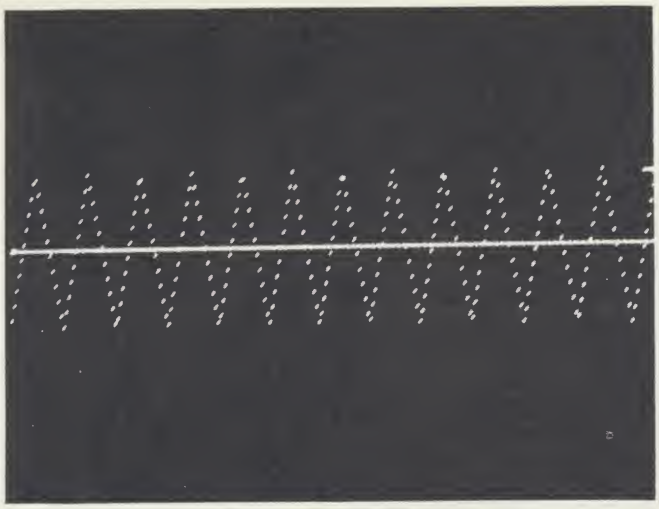
### 3.3.8 Display

All of the spectral programs are written to permit the use of the CRT option for displaying the spectral components and plots. The display program is buffered so that upon call the last preceeding spectrum can be frozen on the display and examined at leisure.

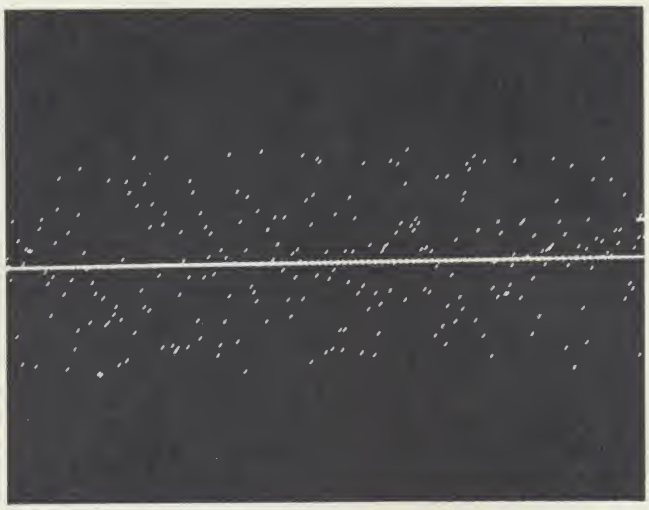
Display options, common to all programs where applicable, include:

- Real part vs imaginary part (phase-plane)
- Real part vs spectral sample number
- Imaginary part vs spectral sample number
- Magnitude vs spectral sample number
- Log-magnitude vs spectral sample number
- Sampled input vs sample number

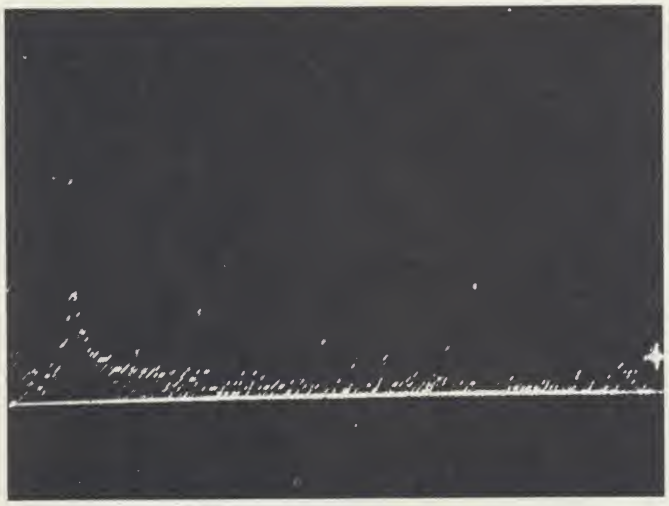
The illustrations on the following pages were taken directly from a CRT on which various functions (which are noted in the captions) were being examined.



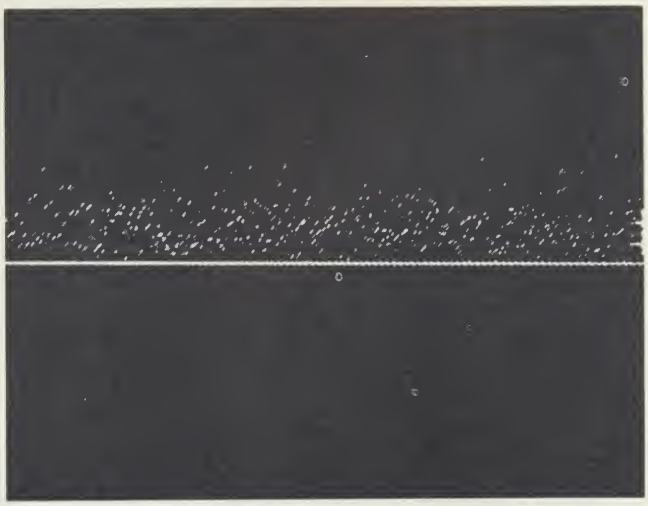
TRIANGULAR WAVE



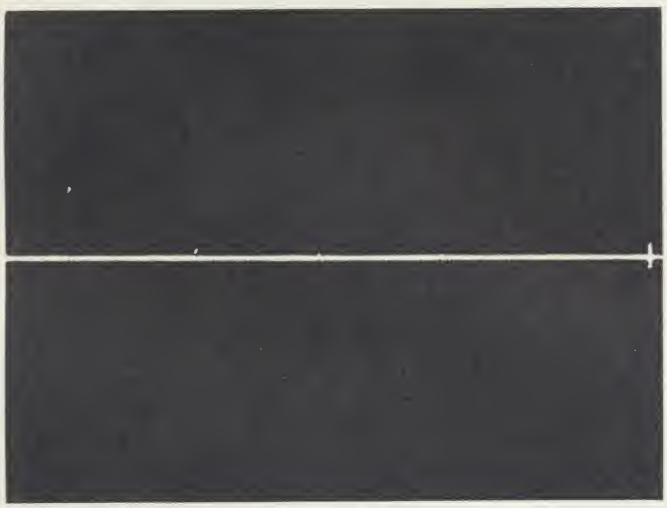
SINEWAVE + GAUSSIAN NOISE



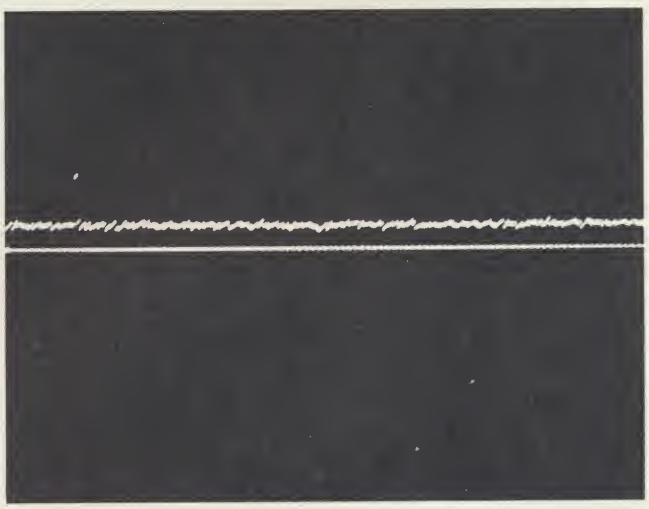
LOG-MAGNITUDE SPECTRUM OF TRIANGULAR WAVE
(7th and 9th harmonics clearly seen.)



THREE SUCCESSIVE SPECTRA
OF SINEWAVE + NOISE

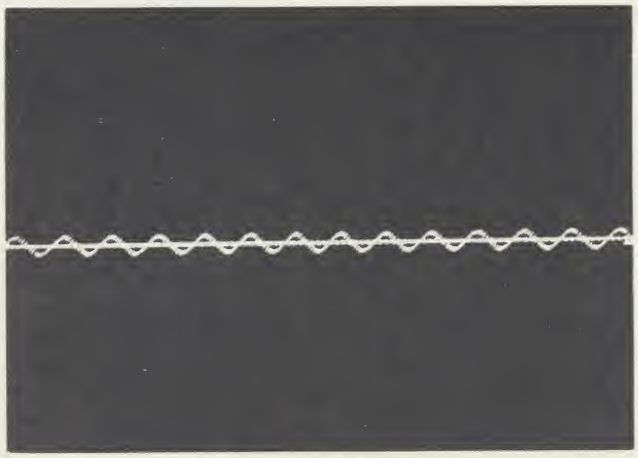


SPECTRUM OF TRIANGULAR WAVE
(Note 7th harmonic hardly visible.)

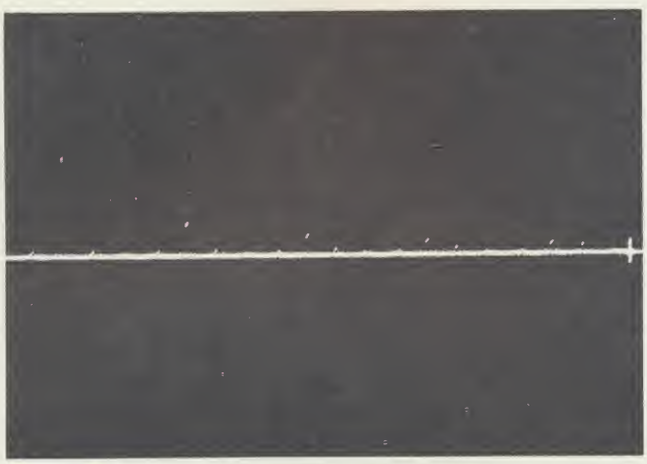


SPECTRUM OF SINEWAVE + NOISE WITH ENSEMBLE AVERAGING
Decrement of $2^{-5}$/spectrum. Note noise level across
the band due to averaging after taking magnitude.

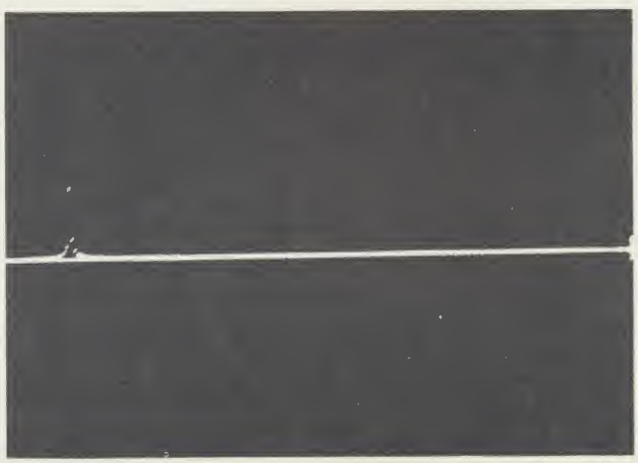
LOG-MAGNITUDE SPECTRUM OF SQUARE WAVE
Folding can be traced beyond 33rd harmonic.



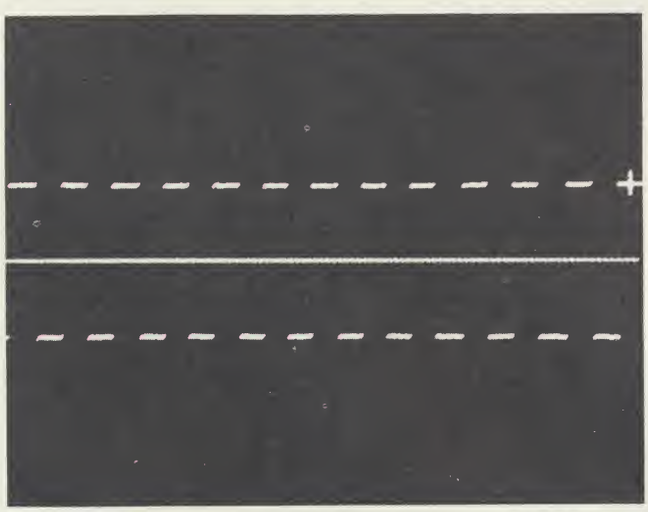
SINEWAVE
(256 samples @ 1 ms/sample)



SPECTRUM OF SQUARE WAVE
(Note folding of 11th and
subsequent harmonies.)



SPECTRUM OF SINEWAVE
(256 samples @ $\frac{500}{256}$ Hz/sample)



SQUARE WAVE

# IV APPLICATIONS

## 4.1 GENERAL COMMENTS

"Applications" is often taken to mean: everything that a machine can be made to do. Unfortunately this definition often includes tasks that are illogical, expensive, or impractical. Ideally, of course, applications are tasks that a machine performs most efficiently. The ideal application also involves as many of the machine's design features as possible.

Many of the CSP-30's more interesting applications stem directly or indirectly from the very high speeds that characterize all facets of its operation and its very design structure: the full cycle time of 100 nanoseconds; the instruction rate of over 3 million instructions per second; and the data acquisition rate of 20 million 8-bit words per second.

Clearly, a major advantage of this speed is that it opens the door to many generic applications and application situations that have great user value. Real time processing of many kinds of data is now possible, and real time processing techniques that were previously either prohibitively costly or actually impossible are now practical.

Multiple, dedicated applications in series are possible, because the CSP-30 is fast enough to compensate for the high cost of time usually required for proper functional programming. The CSP-30's speed also acts to simplify data processing tasks which at present are too clumsy or impractical to be of value.

Finally, the combination of speed and comparative low cost also results in extremely favorable overall levels of cost effectiveness.

A typical generic application is the short duration one-time phenomenon, i.e.; an explosion, a shock, or a burst. Because of the CSP-30's unusually high acquisition rate, data from such occurrences can actually be acquired directly, and very rapidly processed. In many cases, techniques may be applied which allow processing of this data to take place in real time. This capability is clearly of immense value, particularly in such fields as seismic exploration, shock and vibration testing, radar and sonar signal processing, speech research, spectrometry, and production testing.

Typical specific applications are as numerous as there are phenomena to investigate. A small fraction of the thousands of possible applications would include:

- Multi-channel real time seismic data acquisition, and Vibroseis* processing.
- High speed, multi-channel process control.
- Vibration test control and driver signal generation.
- Optical character recognition.
- Sonar doppler signal analysis.
- Communications channel simulation.
- Neuro trace signal analysis and filtering.
- Interferometric (Fourier Transform) and spectral analysis.
- White noise determination of linear and non-linear system response.
- Fine grain doppler radar filtering and analysis.
- Acoustic signature processing.
- Sonar array image structuring.
- Cepstrum vocoder realization.
- Digital filtering; band-pass filtering, recursive digital filtering, high Q

* Vibroseis is a registered trademark of the Continental Oil Company

filtering, low frequency filtering, phase-preserving filters, etcetera.

- o Radar clutter and narrow band doppler filtering.
- o Time waveform generation of specific spectral makeup.
- o Amplitude histogram signal identification.
- o Shock and vibration propagation analysis.
- o General signal analysis: video, audio, facsimilie, radar, sonar, telemetry, etcetera.

## 4.2 SPECIFIC USER APPLICATIONS

### 4.2.1 Background

In mid-1969 the detail specifications and capabilities of the CSP-30 were finalized and assured. At that time it was possible for CSPI's applications specialists to begin evaluating the CSP-30 in terms of the actual requirements of many potential users, whose needs ranged from the simple to the very demanding.

Brief descriptions of a handful of such application situations are provided here. The breadth of these applications is an excellent indication of the flexibility of software-determined computers in general, and the CSP-30 in particular. They also serve as a measure of CSPI's ability to supply creative support in the area of problem analysis, systems analysis, and software.

### 4.2.2 Seismic Exploration

The Vibroseis technique, a method of obtaining seismic and geological strata information independent of the need for drilling dry wells and handling explosives, is rapidly finding favor in the field of seismic exploration. The technique involves the generation of signals whose frequency and time values and relationships are positively known. The returning echoes, carrying changes due to the refractive and reflective characteristics of the earth materials below, are then processed to provide the evaluating geologist with useful data. The technique also improves the possibility of noise reduction over conventional methods.

CSPI's computer system rugged and compact enough to travel into the field with the Vibroseis* equipment, is able to provide the field geoglogist with real time results, and processed test results of very high resolution. This greatly streamlines the entire exploration process and avoids many costly delays.

* Vibroseis is a registered trademark of the Continental Oil Company.

The system uses a ruggedized CSP-30, a ruggedized drum or core mass storage, and appropriate software. It accepts 8000 samples from each of 48 channels at a sampling rate of 500 Hz (2msec/sample), while also generating and sending the pilot signal (a 2 msec/sample stream) to the vibrator. The samples are stored as 12-bit binary numbers, plus 4 bits of a binary gain (SEG format A or B, or any other user-selected format). Successive scans are summed to improve the signal-to-noise ration, after which the summation serves as the input for the Correlation function, and is also permanently stored. Correlation is done with 4000 samples of the pilot signal. The results are available before the next record is begun.

In addition to standard summing and Correlation, the CSPI system does its own formatting, and provides a Vibroseis* monitoring capability previously unavailable.

### 4.2.3 Demodulators

Several programs have been written to demodulate FSK, OOK and PSK signals.

The demodulators are based on band-pass integrate-and-dump filters. Frequencies and baud widths are entered from dial register input devices. The program then looks onto an incoming signal and provides digital outputs under very adverse noise and interference conditions. These programs have been tested on signals with baud width and tone frequency products to 480. The modulation indices of FSK signals (tone separation freqeuncy/baud frequency) have ranged from .35 to 5.

The PSK demodulator has been tested on 2-phase, and 4-phase PSK tones and 2-phase PSK tone stacks. These digital techniques represent almost the only satisfactory demodulation methods for low frequency tones.

These demodulation programs provide display outputs to assist the operator in fine tuning, assure him that the signal is locked on synchronization, provide information on the relative magnitude of FSK tones, and show the number of phases for a PSK tone.

### 4.2.4 A Sixteen-Tone Duplex 4-Phase Modem

This is a large program which accepts input bit stream at the rate of 2400 bits per second and uses these bits to modulate (4-phase DPSK) a group of 16 tones ranging in frequency from 600 Hz to 2200 Hz and which carry the data. A tone at 400 Hz is an empty channel to provide synchronization information.

The program will simultaneously accept samples from the incoming waveform of such a stack, and will demodulate them into an independent 2400 bit/second stream which is then provided an an output to another channel. Finally, the program will detect the frequency shift of the incoming doppler tone, and will route it to an extemal BFO so that the tones can be centered on the proper set of harmonics. The demodulation function employs a modified form of Fast Fourier Transform.

### 4.2.5 Spectrometry

Absorption Spectrometry is one of the analytical chemist's major tools because of its potential for detail and non-destructiveness of samples. Conventional Spectrometry is nevertheless a time consuming process. The advent of Fourier Transform Spectrometry offers a theoretical speed increase of orders of magnitudes particularly for high-resolution spectra.

To provide such high-resolution spectra, CSPI developed a massive Fast Fourier Transform capable of transforming 32,768 sample blocks of 16-bit data.

Many subsidiary problems arise when dealing with a FFT of this size, such as sheer mass of data which must be stored and efficiently manipulated. CSPI's solution was to use a disk memory to store both data and programs, thus making it possible to do the actual calculations using a computer with only 4096 words of memory. Other problems are also handled by the CSPI system. Examples are corrections for instrument errors, signal averaging, control of the instrument and insertion of desired operational parameters.

### 4.2.6 Signal Generation

Although most of CSPI's efforts have been directed to the problems of signal analysis, there have been occasional requirements for assistance in the generation of analog signals. And of course, given certain specifications, such as the magnitude of several spectral components, such a signal can be generated. A digital system employing the FFT algorithm permits precise control over the frequency, amplitude and phase of the spectral components.

4.2.7 Vibration

The vibration testing of complex aerospace systems requires a shake table to be driven with a random signal having a specified spectrum. A technique similar to that described in 4.5.6 is used whereby the spectral magnitudes are specified but the phase of each component is assigned in a psuedo random fashion. Applying the FFT algorithm to this spectrum results in a band limited pseudo random waveform.

The system design may be carried one step further by "closing the loop". If the actual movement of the table is measured and its spectrum analyzed, it can be compared with the desired spectrum. The generated waveform can be modified to make the resultant spectrum match the desired spectrum, thereby correcting for any frequency dependent characteristic of the shake table drive mechanism.

# V INSTRUCTION SET

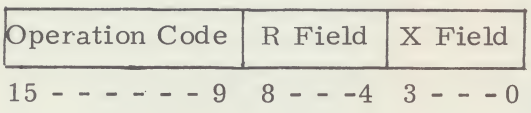
## 5.1 INTRODUCTION

The CSP-30 has an extremely powerful instruction set. It is unusually complete and flexible, having 291 basic instructions (212 one-word, and 79 two-word instructions), many of which have operational variations. The set was designed from a user's standpoint, and with an attention to both scope and detail that has resulted in much greater processing speed than might have been possible, and with an economical compactness of coding.

## 5.2 GENERAL ORGANIZATION

### 5.2.1 Instruction Word Format

The format of the basic instruction set is:




The R (register) field is normally used to select one of the 32 accumulators. The X (index) field specifies which of the fifteen accumulators is to be used as an index register. In 1-word instructions the X field is used to specify an operation on a register or the R and X fields are combined to form: a literal, a short memory address, or an I/O device identification. In 2-word instructions the second word is either immediate data or a memory address. Depending on the instruction, the address may be indexed or indirect.

### 5.2.2 The Instructions

Half of the basic instruction set is made up of arithmetic, logical, and data transfer operations. Eight general operations are included in this group:

Addition
Subtraction
Multiplication
Division
AND
OR
Exclusive OR
Move

Each of these operations has ten variations available. The variations allow the programmer to specify the sources of operands and the destinations of results. These variations are described in detail in later paragraphs.

Four of the ten variations are 1-word instructions. Two select an accumulator, perform any of a number of half-word operations on its contents, and use the result as an operand in the instruction. The other 1-word instructions each use eight bits of the word either as a direct memory address of an operand, or as a literal to be used as an operand.

The other half of the instruction set consists primarily of transfer and test instructions. Many tests are provided in order to minimize programming and execution time. For instance, the contents of two registers may be compared and tested using AND and NOR operations, (other skip conditions include:"less-than", "greater-than", and "equal-to"conditions. Skipping may be initiated on either success or failure of the test.) Other tests that are provided include:

The comparison of a register's contents with zero;

The examination of any bit in an accumulator; and

Status testing: the control flip-flops, the I/O Control Logic Unit, or a peripheral device.

Few jump instructions are necessary because of the variety of skip instructions. The set does provide a 1-word jump instruction using an 11-bit address, and a 2-word jump instruction allowing full memory addressing. A hop instruction is also provided, permitting control transfer to a location relative to the program counter contents. An 8-bit signed number specifies the relationship of the transfer location to the program counter contents.

Pushdown list instructions are included. The contents of a memory address or an accumulator may be added to, or removed from a list in memory. These instructions are also used for subroutine calls and returns. Multiple entry points and recursive subroutines are practical because this method of calling does not destroy the routine's first word. The technique also permits the future design of subroutines in read-only memory.

Logical, arithmetic, long, and short shifts may be performed on the contents of a selected accumulator using the X field of the instruction to specify the number of shifts. Another manipulative instruction enables the programmer to perform simple arithmetic and logical operations on the contents of a chosen register.

The bit patterns of Control instructions determine whether or not certain control flip-flops are to be set, and the indexing, multiplying, or dividing mode to be used.

Additional time-saving data transfer instructions include "exchange", and background Core memory read and write.

Instructions are also provided for incrementing or decrementing the memory by one, or incrementing or decrementing an accumulator by a small literal (+7 to -8).

For each instruction, the listed range of execution times correspond to the number of addressing variations. Computation times are never data-dependent. All execution times discussed in subsequent paragraphs assume the exclusive use of IC memory with a 100-nanosecond cycle time. 900 nanoseconds must be allowed for each complete memory cycle when instructions or operands are located in Core memory. However, in many cases Core memory references will be initiated during idle times, so that writing a word in Core will only delay the instruction execution 100 nsec., and reading a word from Core will only require 400 nanoseconds instead of the full 900. In addition, specialized instructions are provided to perform background transfers of word groups between Core memory and the Accumulator File while the code residing in the IC memory is being executed.

### 5.2.3 Indexing

When indexing is permitted, the contents of the X field determine which register is to be used as an index.

| | |
|---|---|
| X = 0 | No Indexing |
| X=1=to $15_8$ | Use Accumulator X |
| X = $16_8$ | Index by A Register |
| X = $17_8$ | Index by Program Counter |

The normal indexing mode is pre-indexing. The contents of the index register are added to the address word of the instruction. An indirect chain is searched through memory until the indirect bit (15) is a zero. This word is the effective address of Operand 2.

A special post-indexing mode is specified by a control instruction and is in effect for only the next instruction. In this mode one level of indirect addressing is forced before indexing. The contents of the index register are added to this word to form the effective address of Operand 2.

## 5.3 ARITHMETIC AND LOGICAL INSTRUCTIONS

The instructions in this group are:

Add

Subtract

Multiply

Divide

Logical Inclusive OR

Logical Exclusive OR

Logical AND

Each of these instructions has two 16-bit operands and a 16-bit result. (some multiply and divide modes provide a 32-bit result). Ten basic operand addressing variations (some with sub-variations) are provided; these apply identically to all seven members of the arithmetic/logical group.

The following statements apply to all arithmetic/logical instructions:

1. Operand one is always the contents of a register in the accumulator file. Except in variation eight, for which the A register is implied, any of RØ - R37 may be selected. RØ is the A register.

2. Operand two is derived in one of several different ways, determined by the variation chosen:

   a. contents of the A register (RØ).

   b. eight-bit literal (-128 to +127).

   c. eight-bit direct memory address (0 to $377_8$).

   d. full-word immediate; may be indexed.

   e. full-word memory address; indexing and/or indirect addressing permitted.

3. The results are always placed in the A register (RØ).

4. In addition, the result may be stored in one of several other places, determined by the variation chosen:

   a. nowhere (result placed only in A).

   b. accumulator file register designated as operand one.

   c. memory location addressed by second word of instruction, with indexing and/or indirect addressing permitted.

5.3.1 Addressing Variations

Below are described the eight addressing variations for the arithmetic/logical instructions. To the execution times listed, add 8 cycles for multiply instructions and 35 cycles for divide instructions.

1. Instruction length: 1 word
   Operand 1: Any R, with optional halfword manipulation (see below)
   Operand 2: A
   Result Destination: Just A, or A and R --depends on high-order bit in X field.
   Execution Time: 2 cycles (3 if result put into both A and R).

For this variation only, the low-order three bits of the X field allow certain permutations to be applied to operand 1 prior to calculation of the result. (The contents of R are not altered by this.) The following manipulations are permitted.

   First: a. leave operand unchanged, or
          b. swap half-words

   Then: a. leave operand unchanged, or
         b. zero low-order half, or
         c. zero high-order half, or
         d. sign-extend low-order half.

2. Length: 2 words
   Operand 1: Any R
   Operand 2: Immediate; indexing allowed.
   Destination: A and R
   Time: 3 cycles, 5 with indexing

3. Length: 2 words
   Operand 1: Any R
   Operand 2: Memory; indexing and/or indirect addressing allowed.
   Destination: A and R
   Time: 4 cycles, plus 1 if indexing, plus 1 for each level of indirect addressing.

4. Length: 2 words
   Operand 1: Any R
   Operand 2: Memory; indexing and/or indirect addressing allowed.
   Destination: A and memory (same effective address as Operand 2).
   Time: 5 cycles, plus 1 if indexing, plus 1 for each level of indirect addressing.

5. Length: 2 words
   Operand 1: Any R
   Operand 2: A
   Destination: A and memory; indexing and/or indirect addressing allowed.
   Time: 5 cycles, plus 1 if indexing plus 1 for each level of indirect addressing.

6. Length: 2 words
   Operand 1: Any R
   Operand 2: Memory; indexing and/or indirect addressing allowed
   Destination: A only
   Time: 4 cycles, plus 1 if indexing, plus 1 for each level of indirect addressing.

7. Length: 2 words
   Operand 1: Any R
   Operand 2: Immediate; indexing allowed
   Destination: A only
   Time: 3 cycles, 5 with indexing.

8. Length: 1 word
   Operand 1: A
   Operand 2: Half-word literal, or memory with 8-bit direct address.
   Destination: A only
   Time: 2 cycles for literal, 5 cycles for memory.

Variation eight includes two sub-variations for determining operand two, distinguished by bit 8 of the instruction. If this bit is reset, then bit 7 is complemented and sign-extended to form a literal (-128 tc +127). Otherwise, the low-order half-word is taken as a direct memory address, (in the range of 000 to $377_8$).

### 5.3.2 Multiply/Divide Features

Control flip-flops condition the detailed behavior of both the multiply and divide instructions. Each instruction may operate in either integer or fraction mode. (In fraction format, the "binary point" is immediately to the right of the sign bit.) In addition, each may store either one word or two words of result. The two-word result of a multiplication is a 32-bit number (left-adjusted and zero-filled if a fraction, right-adjusted and sign-extended if an integer). The low-order word is placed in the A register and the high-order word is placed in whatever other destination is indicated by the particular addressing mode chosen. If only one word is to be stored, then it is the high-order (rounded) for fractional multiplication and the low-order word for integer multiplication; in either case, the single-word result is placed in all indicated destinations. The two-word result of a division consists of a quotient word and a remainder word. If both are to be stored, then the remainder is placed in A and the quotient in whatever other destination is selected. If only one word is to be stored, this word is the (rounded for fractional divide) quotient, and it is placed in all specified destinations.

## 5.4 INCREMENT MEMORY WORD INSTRUCTIONS

The contents of the designated memory location are increased by one. The two available addressing options for these instructions are given below:

1. Instruction Length: 2 words
   Operand: Any memory word; optional indexing and/or indirect addressing.
   Time: 5 cycles, plus 1 if indexing, plus 1 for each level of indirect addressing.

2. Instruction Length: 1 word
   Operand: Memory word (Ø to $377_8$)
   Time: 6 cycles

## 5.5 DATA TRANSFER INSTRUCTIONS

The "moves" are conceptually members of the arithmetic/logical group by virtue of their operand addressing modes. They are treated separately here because their single argument takes them out of the pattern of arithmetic instructions.

Addressing variations permitted are:

1. Move Register to A; Move A to Register

| | |
|---|---|
| Instruction Length: | 1 word |
| Data Source: | Case 1, any R; case 2, A. Half-word function permitted. |
| Data Destination: | Case 1, A; case 2, any R |
| Time: | Case 1, 2 cycles; case 2, 3 cycles. |

The high-order bit of the X field distinguishes two sub-variations. In one case, the contents of R are moved to A; in the other, the contents of A are moved to R. The low-order 3 bits of the X field specify a half-word permutation to be applied to the data before it is stored, as in a variation of the arithmetic/logical instructions. (See Subsection 5.2.1).

2. Move Immediate to Register

| | |
|---|---|
| Length: | 2 words |
| Source: | Immediate, indexing allowed |
| Destination: | Any R |
| Time: | 3 cycles, plus 1 if indexing. |

3. Move Memory to Register

| | |
|---|---|
| Length: | 2 words |
| Source: | Memory, indexing and/or indirect addressing allowed. |
| Destination: | Any R |
| Time: | 4 cycles, plus 1 if indexing, plus 1 for each level of indirect addressing. |

4. Move Register to Memory

| | |
|---|---|
| Length: | 2 words |
| Source: | Any R |
| Destination: | Memory; indexing and/or indirect addressing allowed. |
| Time: | 5 cycles, plus 1 if indexing, plus 1 for each level of indirect addressing. |

5. Move Zero to Memory

| | |
|---|---|
| Length: | 2 words |
| Source: | None: Zero is stored. |
| Destination: | Memory; indexing and/or indirect addressing allowed. |
| Time: | 4 cycles, plus 1 if indexing, plus 1 for each level of indirect addressing. |

6. Move Literal to A; Move memory (8-bit direct address) to A

| | |
|---|---|
| Length: | 1 word |
| Source: | Half-word immediate, or else memory with 8-bit direct address. |
| Destination: | A |
| Time: | 2 cycles for immediate, 4 cycles for memory. |

Operand addressing for the above instructions (1 through 6) parallels that of arithmetic/logical instructions (see Subsection 5.2.1).

7. Move A to Memory (8-bit direct Address

| | |
|---|---|
| Length: | 1 word |
| Source: | A |
| Destination: | Memory word located by 8-bit direct address |
| Time: | 5 cycles |

8. Move Zero to Memory (8-bit direct Address)

| | |
|---|---|
| Length: | 1 word |
| Source: | None: Zero is stored |
| Destination: | Memory word located by 8-bit direct address |
| Time: | 5 cycles |

9. Exchange Memory and Register

| | |
|---|---|
| Length: | 2 words |
| Time: | 5 cycles, plus 1 if indexing plus 1 for each level of indirect addressing |

10. Background Core Memory Read

| | |
|---|---|
| Length: | 2 words |

Source: Any two consecutive core locations, indexing and/or indirect addressing allowed. (A special mode allows the transfer of 4, 8, or 16 words).

Destination: Any two (4, 8, 16) consecutive accumulator file registers, beginning at an even-numbered register $\geq$ R-16

Time: 3 cycles, plus 2(4, 8,16) "stolen" later, plus 1 if indexing, plus 1 for each level of indirect addressing.

This special-purpose instruction is used to initiate access to data residing in Core Memory, somewhat in advance of the time when it actually is needed for calculation. Two consectuive words from Core Memory are copied into two consecutive Accumulator File registers.(If reference to the data is attempted before the access is completed, a normal memory access will be imposed until the data is ready). With skillful program organization, this instruction ordinarily will eliminate most of the latency time spent waiting for operands from Core Memory.

11.Background Core Memory Write

Length: 2 words

Source: Any two consecutive accumulator file registers, beginning at an even-number register. R16.

Destination: Any two consecutive core locations, beginning at an even-number location; indexing and/or indirect addressing allowed.

Time: 3 cycles, plus 1"stolen" later, plus 1 if indexing, plus 1 for each level of indirect addressing.

## 5.6 OPERATIONS ON ACCUMULATOR FILE REGISTERS

1. Increment Register by Small Literal

Length: 1 word

Operand 1: Any R

Operand 2: Literal in X field, (-8 to +7).

Destination: R

Time: 3 cycles

Bit 3 of the X field is complemented and sign-extended to form a literal, (in the range of -8 to +7). This quantity is added to the contents of the register specified, and the result is placed in this register.

2. Elementary Arithmetic/Logical Operations

Length: 1 word

Source: Any R

Destination: A, or A and R (conditioned by high-order bit of X field).

Time: 3 cycles.

One of eight operations is performed on the contents of R;

a. Zero (all bits clear)
b. Minus one (clear and complement)
c. One's complement
d. Negate (two's complement)
e. Complement if negative (one's-complete magnitude)
f. Negate if negative (two's complement magnitude).
g. Add contents of "carry" flip-flop (facilitates multiple-precision addition).
h. Add contents of "carry" flip-flop minus one (facilitates multiple-precision subtraction).

3. Shift Instructions

Length: 1 word

Operand: Any R, or else any R in combination with A.

Time: 3 cycles, plus shift count.

An arbitrary accumulator file register is shifted--either left or right, either arithmetically or logically, and either single or in combination with the A register. In double-words shifts, the A register is the low-order word.) Vacated bits at the left are filled with copies of the sign bit during arithmetic right shifts and with zeros during logical right shifts; there is no distinction between arithmetic and logical left shifts: vacated bits at the

right are always filled with zeros. The shift distance, 0 to 15 bit positions, is specified by the 4-bit X field in the instruction. Note that, since the A register is conceptually RØ of the accumulator file, a logical left double-word shift in which RØ is selected serves to rotate the A register to left.

A Normalize instruction preceeding a shift left $17_8$ will cause it to terminate with the most significant bit of the word immediately to the right of the sign bit and the number of shifts executed is placed in the A register.

## 5.7 SKIP INSTRUCTIONS

Program branching is accomplished by means of conditional skip instructions. Each such instruction performs a particular logical test on its operand(s). If this test succeeds the next whole instruction in sequence is ignored and the program counter is advanced the appropriate number (one or two) of locations. If the test fails, the next instruction is executed and the instruction sequence is unaltered. Ordinarily, a section of program to be executed conditionally is entered by a jump instruction following an appropriate skip instruction. In many cases, however, the requisite section of program will consist of a single instruction, (e.g., a subroutine call) which itself may be placed after a skip instruction; moreover, compound logical tests may be performed by a "nest" of skip instructions--i.e., two or more skip instructions in sequence.

### 5.7.1 Two-Operand Arithmetic/Logical Comparisons

This instruction group provides five common arithmetic/logical comparisons and their inverses. Operand one (denoted R) is the contents of arbitrary member of the accumulator file; operand two (denoted I/M) is either immediate data (with indexing allowed) or else data from memory (with indexing and/or indirect addressing allowed). These are all two-word instructions. Execution time listed assumes operand two is immediate data. Add 1 cycle if operand two is in memory; add 1 cycle if indexing; add 1 cycle for each level of indirect addressing; add 1 cycle for each location skipped when the test succeeds.

The following comparisons are available. A Skip occurs if the comparison is true; if the comparison is false the program continues in sequence.

| | | |
|---|---|---|
| R = I/M | R ≠ I/M | 4 cycles |
| R > I/M | R < I/M | 3 cycles |
| R ≤ I/M | R ≥ I/M | 3 cycles |
| R ∧ I/M = 0 | R ∧ I/M ≠ 0 | 4 cycles |
| R ∨ I/M = 1 | R ∨ I/M = -1 | 4 cycles |

The last two instructions may be used to test a combination of bits being used as flags for various conditions.

### 5.7.2 One-Operand Tests

These are all one-word instructions. Execution time is three cycles, plus one cycle for each location skipped when the test succeeds.

#### 5.7.2.1 Skip If Bit Set Reset

An arbitrary bit of an arbitrary Accumulator File register is addressed. Two tests are permitted:

1. Skip if bit X of register R is set.
2. Skip if bit X of register R is reset.

#### 5.7.2.2 Skip If Condition True/False

These instructions permit certain arithmetic/logical tests to be performed on the contents of an arbitrary member of the Accumulator File. The tests are:

| | |
|---|---|
| R = 0 | R ≠ 0 |
| R = -1 | R ≠ -1 |
| R ≤ 0 | R > 0 |
| R ≥ 0 | R < 0 |

In addition, these instructions are used to detect certain conditions within the processor and the I/O system. These are:

| | |
|---|---|
| Carry flop set | Carry flop reset |
| Overflow flop set | Overflow flop reset |
| Divide-check flop set | Divide-check flop reset |
| Misc. I/O tests true | Misc. I/O tests reset |

## 5.8 JUMP INSTRUCTIONS

The following Jump instructions are available.

### 5.8.1 Unconditional Jump

Length: 2 words
Time: 2 cycles, plus 1 if indexing, plus 1 for each level of indirect addressing.

Program control is transferred to the location addressed by the second word of the instruction. Indexing and/or indirect addressing are permitted.

### 5.8.2 Unconditional Jump To 11-Bit Absolute Address

Length: 1 word
Time: 2 cycles

Program control is transferred to the location addressed by the low-order eleven bits of the instruction, (range of 0000 to $3777_8$).

### 5.8.3 Unconditional Short Jump Relative to PC

Length: 1 word
Time: 3 cycles

Bit 9 of the instruction is complemented and sign extended to form a number, (range of $-400_8$ to $377_8$), which is added to the contents of the program counter. The result of the addition is placed in the program counter, transferring program control to this new location.

### 5.8.4 Loop-Terminating Instructions

Length: 2 words
Time: 3 cycles, plus 1 for each level of indirect addressing

The contents of an arbitrary accumulator file register are incremented by a number (-8 to +7). A decision is made whether or not to jump, conditioned by the new contents of the register. Either "jump if positive" or "jump if negative" may be used. The effective jump address is determined by the second word of the instruction, with indirect addressing allowed (but not indexing).

## 5.9 SUBROUTINE CALLS AND PUSH-DOWN-LIST INSTRUCTIONS

Each of these instructions modifies a list of memory, as well as a pointer to this list contained in an arbitrary member of the accumulator file. This accumulator addresses the most recent entry (highest-numbered memory word) in the list, and a word is added to the list by incrementing the pointer and then storing into the new location it addresses. A word is read (and removed) from the list by fetching it from the memory location addressed by the list pointer and then decrementing the pointer. A list pointer is always interpreted as a direct address and its sign bit is ignored.

### 5.9.1 Subroutine Call (2-Word)

Length: 2 words
List Pointer: Any R
List Element Source: Program Counter
Time: 4 cycles, plus 1 if indexing, plus 1 for each level of indirect addressing.

The contents of the program counter (address following that of the second word of the instruction) are placed on the list addressed by R. Program Control is transferred to the address specified by the second word of the instruction (with indexing and/or indirect addressing allowed), by placing the effective address in the program counter.

### 5.9.2 Return From Subroutine

Length: 1 word
List Pointer: Any R
List Element Destination: Program Counter
Time: 4 cycles

A word is removed from the list addressed by R and is placed in the program counter.

### 5.9.3 Push Register Onto List

Length: 1 word
List Pointer: Any of RØ-15
List Element Source: Any R
Time: 5 cycles

### 5.9.4 Pop To Register From List

| | |
|---|---|
| Length: | 1 word |
| List Pointer: | Any of RØ-15 |
| List Element Destination: | Any R |
| Time: | 5 cycles |

### 5.9.5 Push Memory Onto List

| | |
|---|---|
| Length: | 2 words |
| List Pointer: | Any R |
| List Element Source: | Memory, with indexing and/or indirect addressing allowed. |
| Time: | 5 cycles, plus 1 if indexing, plus 1 for each level of indirect addressing. |

### 5.9.6 Pop To Memory From List

| | |
|---|---|
| Length: | 2 words |
| List Pointer: | Any R |
| List Element Destination: | Memory, with indexing and/or indirect addressing allowed. |
| Time: | 5 cycles, plus 1 if indexing, plus 1 for each level of indirect addressing. |

### 5.9.7 Interrupt

| | |
|---|---|
| Length: | 1 word |
| List Pointer: | Special Register |
| List Element Destination: | Program Counter, A Register, Program Control Word. |
| Time: | Multi level interrupt processing is facilitated by automatically storing all registers pertaining to the machine status on a push down list. The amount of storage reserved for this list is the only limit on the depth of interrupt nesting permitted. |

The pointer to the interrupt list is address Ø in the Accumulator File. It is masked by the A register for all other instructions.

## 5.10 INPUT/OUTPUT INSTRUCTIONS

The basic I/O instructions are Connect Input, Connect Output, and I/O Command. The Connect Input instruction causes a word to be read from an external device. This word may be data or sense information for a particular device or interrupt enable/request information from each device or a channel. The word is used as an operand in the next instruction. The Connect Output instruction allows the result of an instruction to be transferred to an external device. In addition to direct data transfers, the CSP-30 I/O Command Instruction transfers a 3-bit function code to an external device.

The I/O can be grouped in two categories: data transfer commands, and control commands. The first category affects only the source of the operand or destination of the result of the next instruction executed by the CPU. Each instructions, which cause an input word from a peripheral, add one cycle to the execution time of the next instruction. The second category of instruction transfers control information from the CPU to the peripheral. Such I/O Command Insturctions are stand-alone instructions and require only one cycle.

# NOTES

209 Middlesex Turnpike, Burlington, Massachusetts 01803 (617) 272-6020

020202